1. Program to create a window with different colours using OpenGL.

#include<GL/glu.h>

#include <GL/glut.h>

void MyInit()

{

glClearColor(0,0,1,1);

}

void draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glBegin(0);

glEnd();

glFlush();

}

int main(int c, char \*v[])

{

glutInit(&c,v);

glutInitWindowPosition(300,300);

glutInitWindowSize(400,400);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutCreateWindow("My first window");

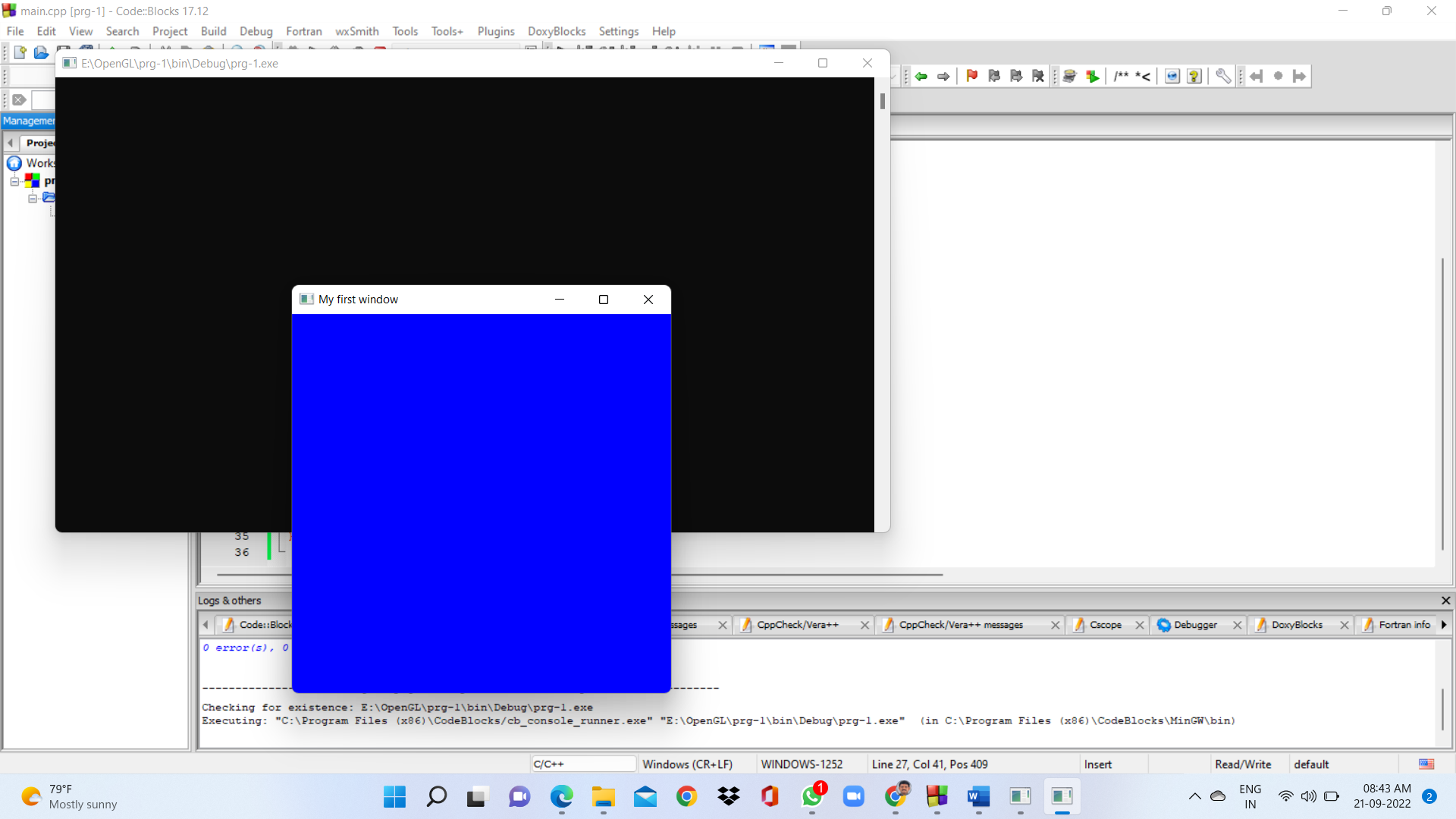
MyInit();

glutDisplayFunc(draw);

glutMainLoop();

return 0;

}



2. Program to create a window with four vertices using OpenGL.

#include<GL/glu.h>

#include <GL/glut.h>

void MyInit()

{

glClearColor(0,0,1,1);

glColor3f(1,0,0);

}

void draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glPointSize(5);

glBegin(GL\_POINTS);

glVertex2f(-0.6,0.6);

glVertex2f(0.6,0.6);

glVertex2f(0.6,-0.6);

glVertex2f(-0.6,-0.6);

glEnd();

glFlush();

}

int main(int c, char \*v[])

{

glutInit(&c,v);

glutInitWindowPosition(300,300);

glutInitWindowSize(400,400);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutCreateWindow("My first window");

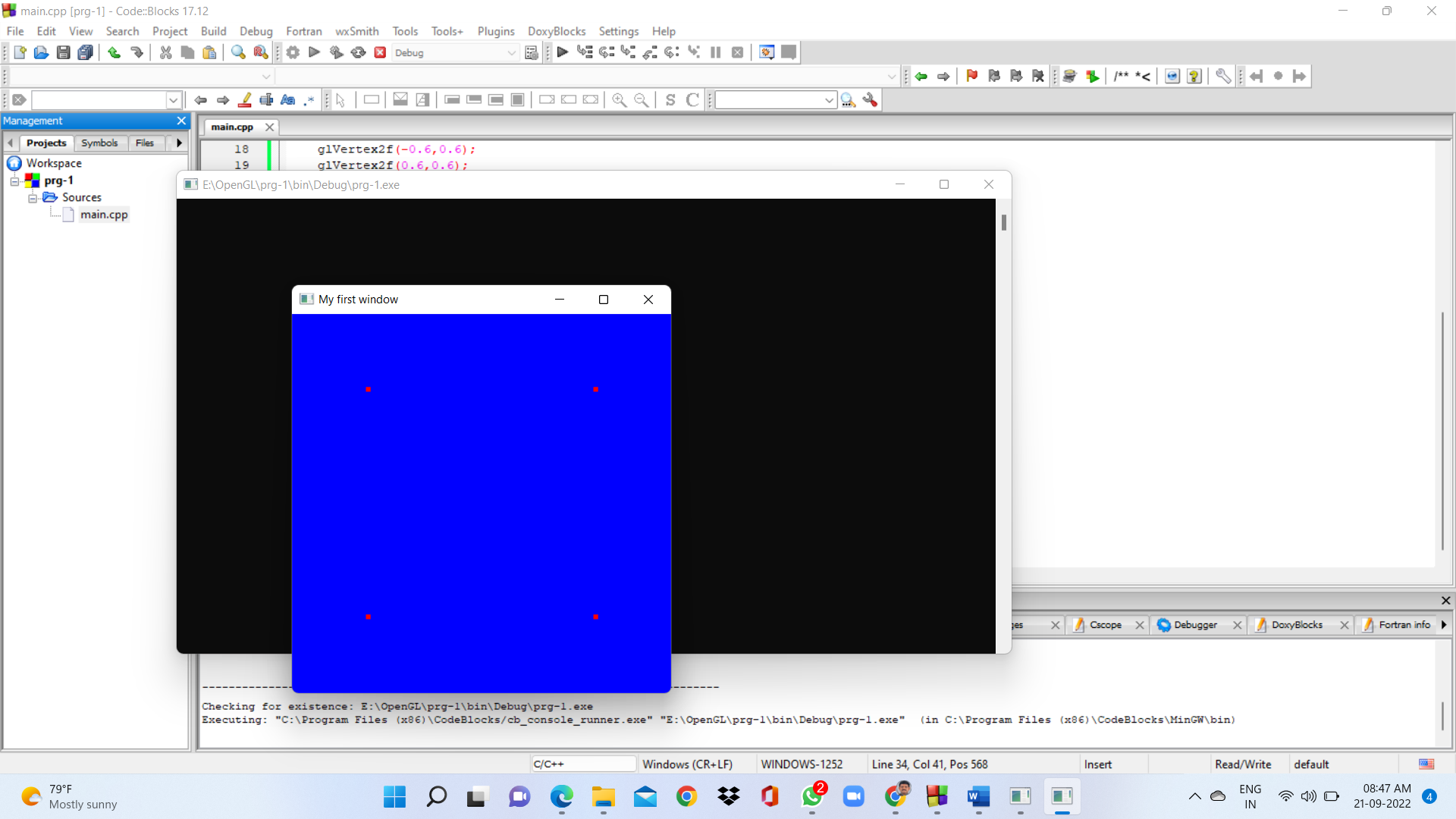
MyInit();

glutDisplayFunc(draw);

glutMainLoop();

return 0;

}



3. Program to draw two parallel lines in a window using OpenGL.

#include<GL/glu.h>

#include <GL/glut.h>

void MyInit()

{

glClearColor(0,0,1,1);

glColor3f(1,0,0);

}

void draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glPointSize(5);

glBegin(GL\_LINES);

glVertex2f(-0.6,0.6);

glVertex2f(0.6,0.6);

glVertex2f(0.6,-0.6);

glVertex2f(-0.6,-0.6);

glEnd();

glFlush();

}

int main(int c, char \*v[])

{

glutInit(&c,v);

glutInitWindowPosition(300,300);

glutInitWindowSize(400,400);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutCreateWindow("My first window");

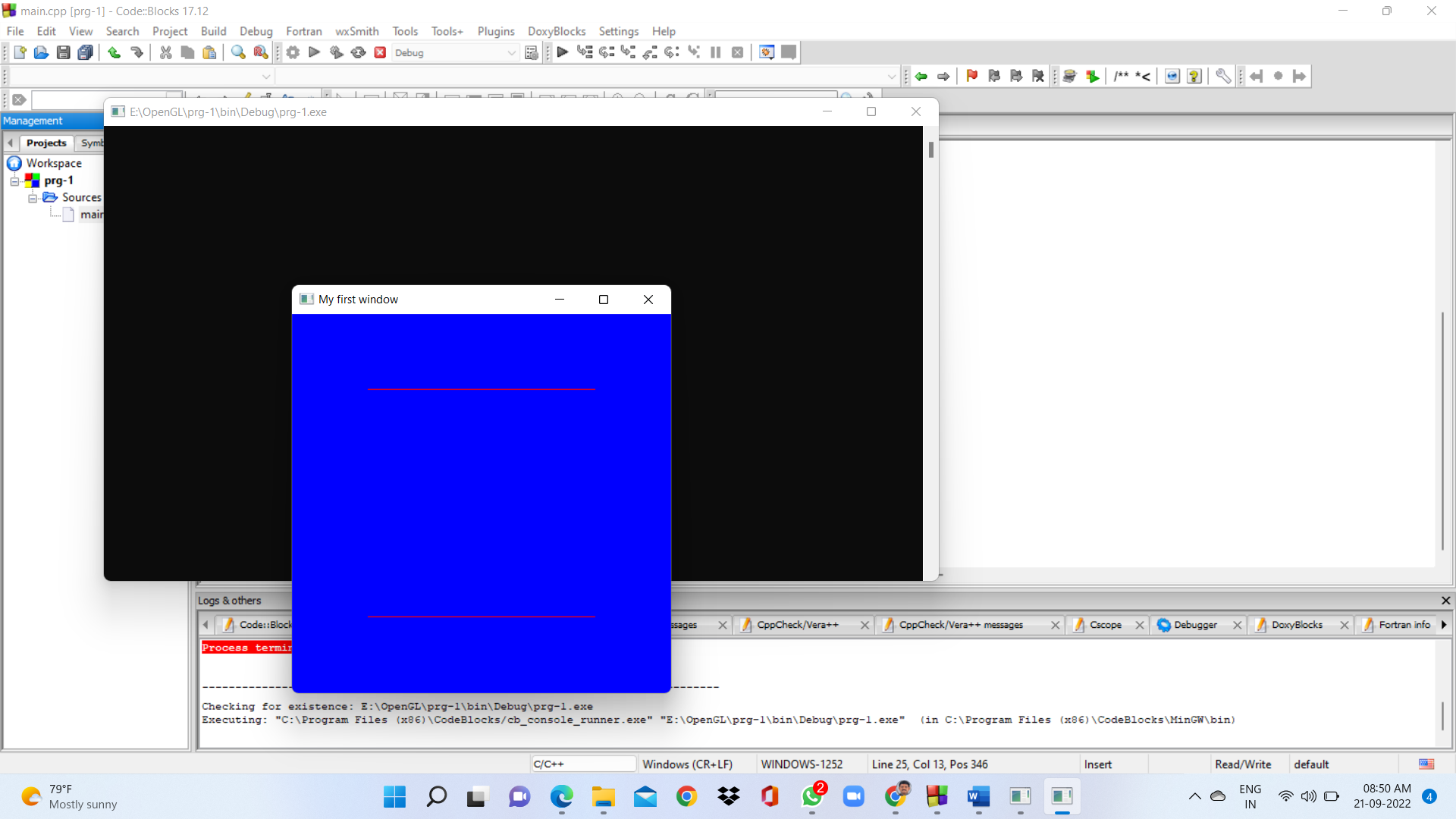
MyInit();

glutDisplayFunc(draw);

glutMainLoop();

return 0;

}



4. Program to draw a square in a window using OpenGL.

#include<GL/glu.h>

#include <GL/glut.h>

void MyInit()

{

glClearColor(0,0,1,1);

glColor3f(1,0,0);

}

void draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glPointSize(5);

glBegin(GL\_LINE\_LOOP);

glVertex2f(-0.6,0.6);

glVertex2f(0.6,0.6);

glVertex2f(0.6,-0.6);

glVertex2f(-0.6,-0.6);

glEnd();

glFlush();

}

int main(int c, char \*v[])

{

glutInit(&c,v);

glutInitWindowPosition(300,300);

glutInitWindowSize(400,400);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutCreateWindow("My first window");

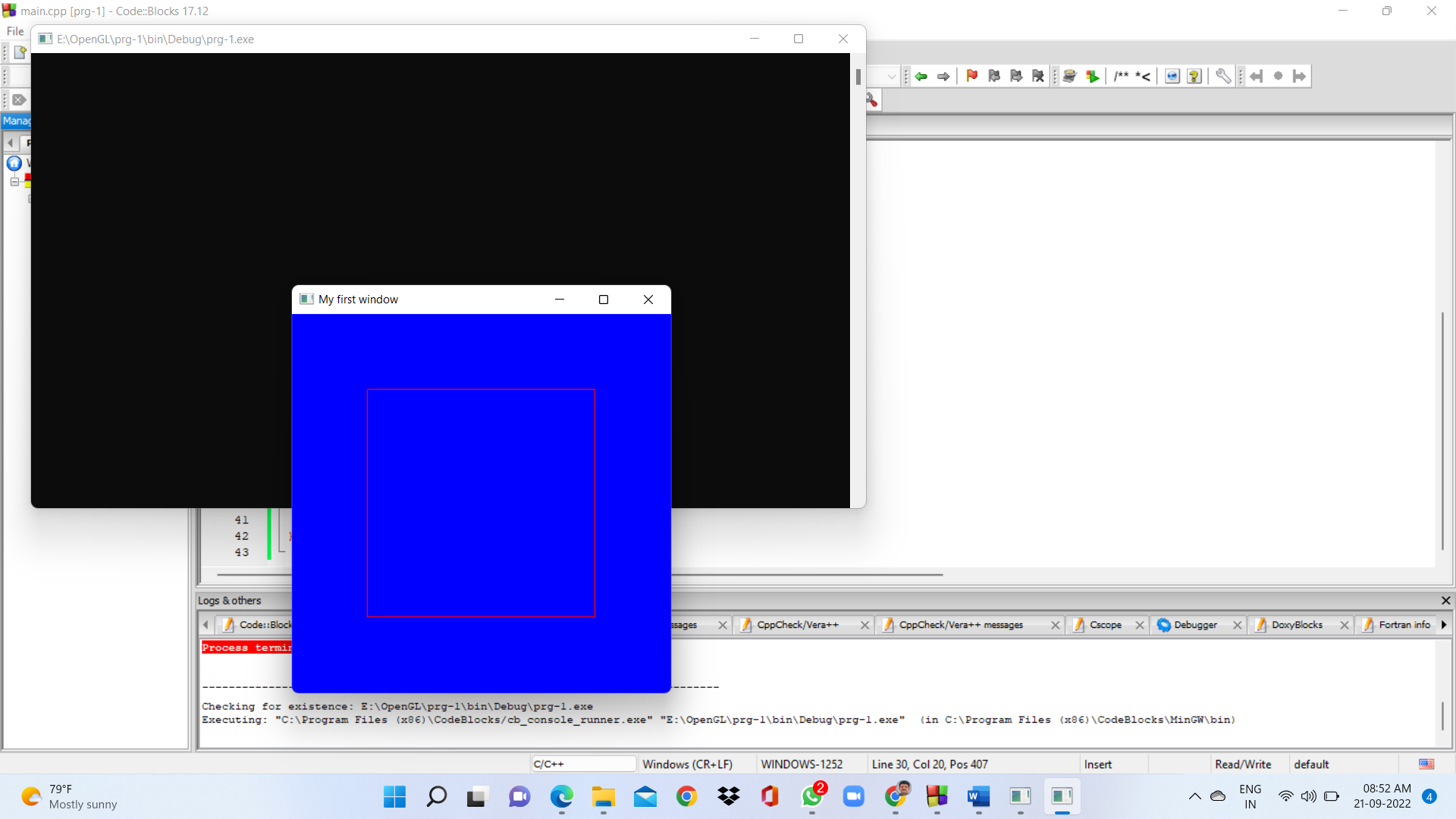
MyInit();

glutDisplayFunc(draw);

glutMainLoop();

return 0;

}



5. Program to draw a rectangle in a window using OpenGL.

#include<GL/glu.h>

#include <GL/glut.h>

void MyInit()

{

glClearColor(0,0,1,1);

glColor3f(1,0,0);

}

void draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glPointSize(5);

glBegin(GL\_LINE\_LOOP);

glVertex2f(-0.6,0.6);

glVertex2f(0.6,0.6);

glVertex2f(0.4,-0.6);

glVertex2f(-0.8,-0.6);

glEnd();

glFlush();

}

int main(int c, char \*v[])

{

glutInit(&c,v);

glutInitWindowPosition(300,300);

glutInitWindowSize(400,400);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutCreateWindow("My first window");

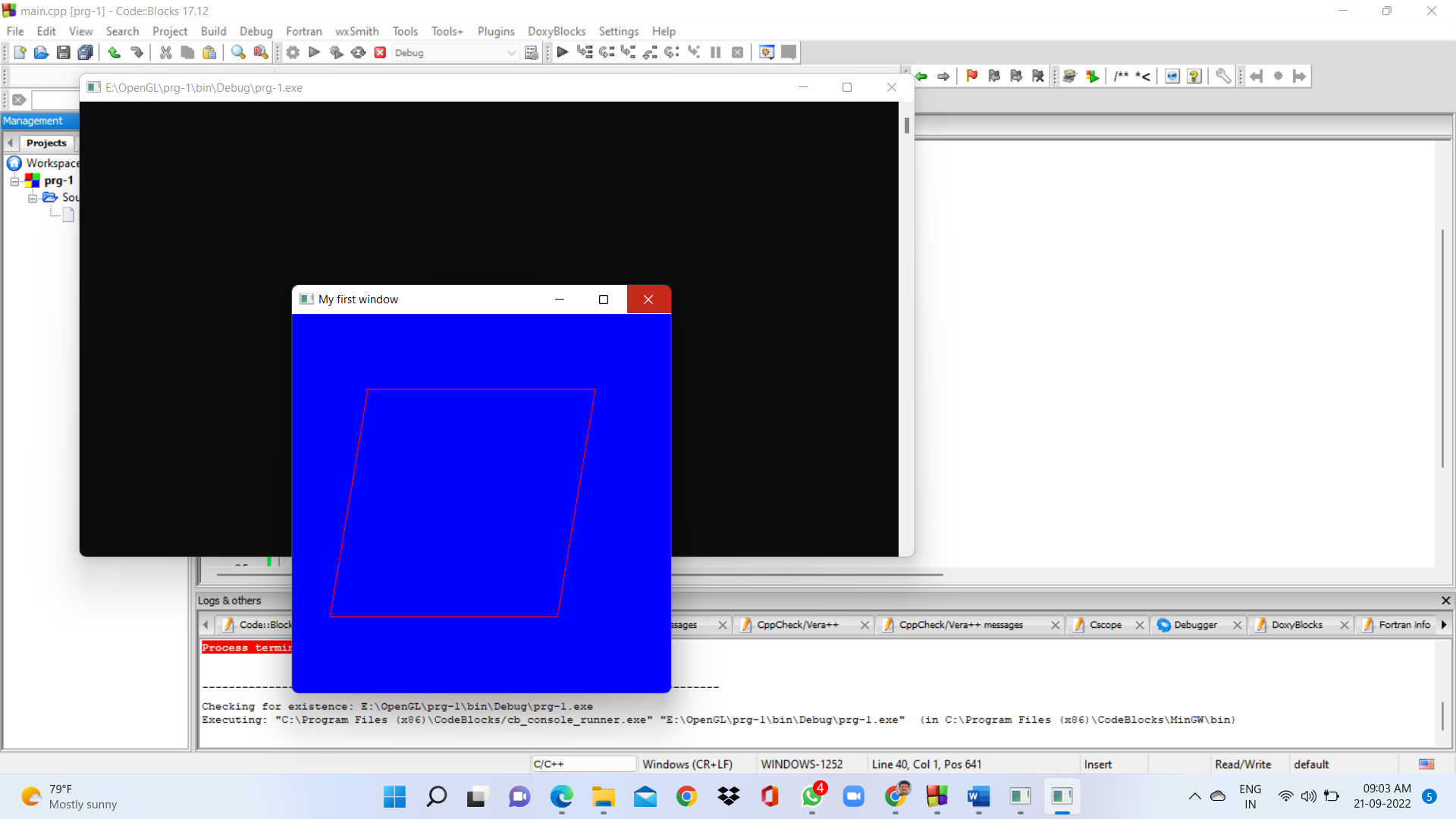
MyInit();

glutDisplayFunc(draw);

glutMainLoop();

return 0;

}



6. Program to draw a connected group of line segments with RED colour from first vertex to last.

#include<GL/glu.h>

#include <GL/glut.h>

void MyInit()

{

glClearColor(0,0,1,1);

glColor3f(1,0,0);

}

void draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glPointSize(5);

glBegin(GL\_LINE\_STRIP);

glVertex2f(-0.6,0.6);

glVertex2f(0.6,0.6);

glVertex2f(0.4,-0.6);

glVertex2f(-0.8,-0.6);

glEnd();

glFlush();

}

int main(int c, char \*v[])

{

glutInit(&c,v);

glutInitWindowPosition(300,300);

glutInitWindowSize(400,400);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutCreateWindow("My first window");

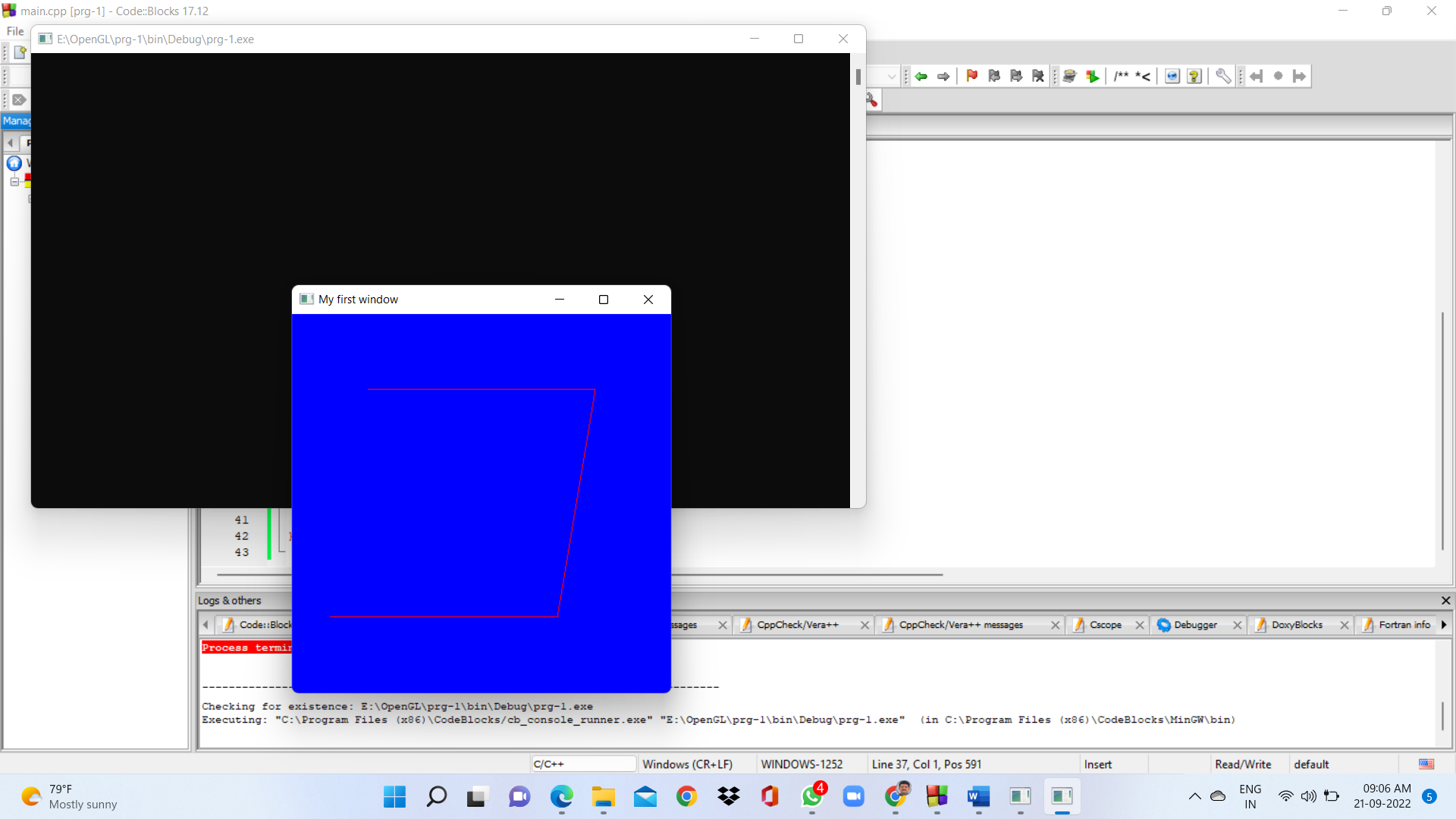
MyInit();

glutDisplayFunc(draw);

glutMainLoop();

return 0;

}



7. Program to draw a triangle with GREEN colour in a window using OpenGL.

#include<GL/glu.h>

#include <GL/glut.h>

void MyInit()

{

glClearColor(0,0,1,1);

glColor3f(1,0,0);

}

void draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glPointSize(5);

glBegin(GL\_LINE\_LOOP);

glVertex2f(0.0,0.6);

glVertex2f(0.8,-0.6);

glVertex2f(-.8,-0.6);

glEnd();

glFlush();

}

int main(int c, char \*v[])

{

glutInit(&c,v);

glutInitWindowPosition(300,300);

glutInitWindowSize(400,400);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutCreateWindow("My first window");

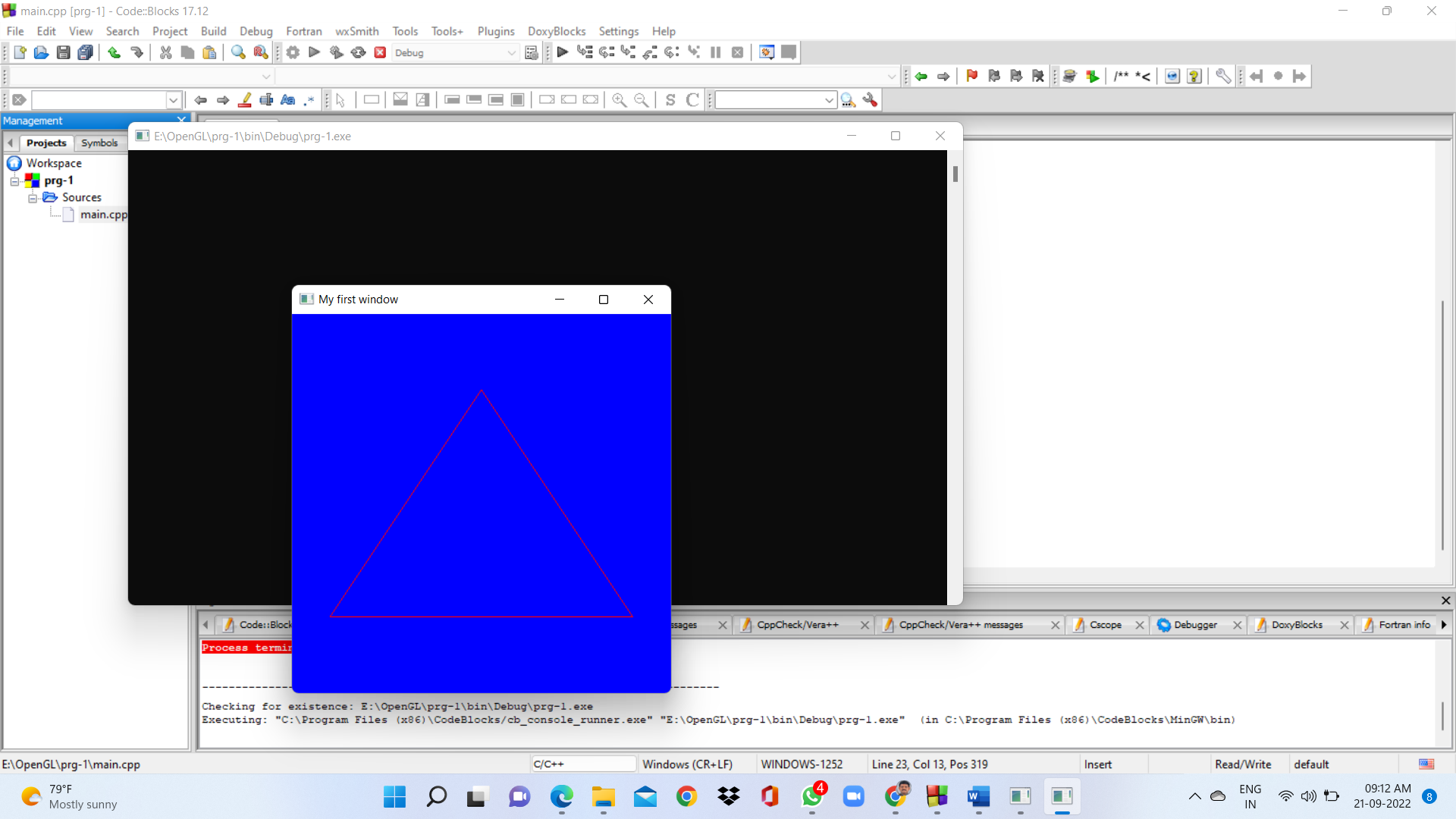
MyInit();

glutDisplayFunc(draw);

glutMainLoop();

return 0;

}



8. Program to draw a polygon in a newly created window using OpenGL.

#include<GL/glu.h>

#include <GL/glut.h>

void MyInit()

{

glClearColor(0,0,1,1);

glColor3f(1,0,0);

}

void draw()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glPointSize(5);

glBegin(GL\_POLYGON);

glVertex2f(0.0,0.6);

glVertex2f(0.8,-0.6);

glVertex2f(-.8,-0.6);

glEnd();

glFlush();

}

int main(int c, char \*v[])

{

glutInit(&c,v);

glutInitWindowPosition(300,300);

glutInitWindowSize(400,400);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutCreateWindow("My first window");

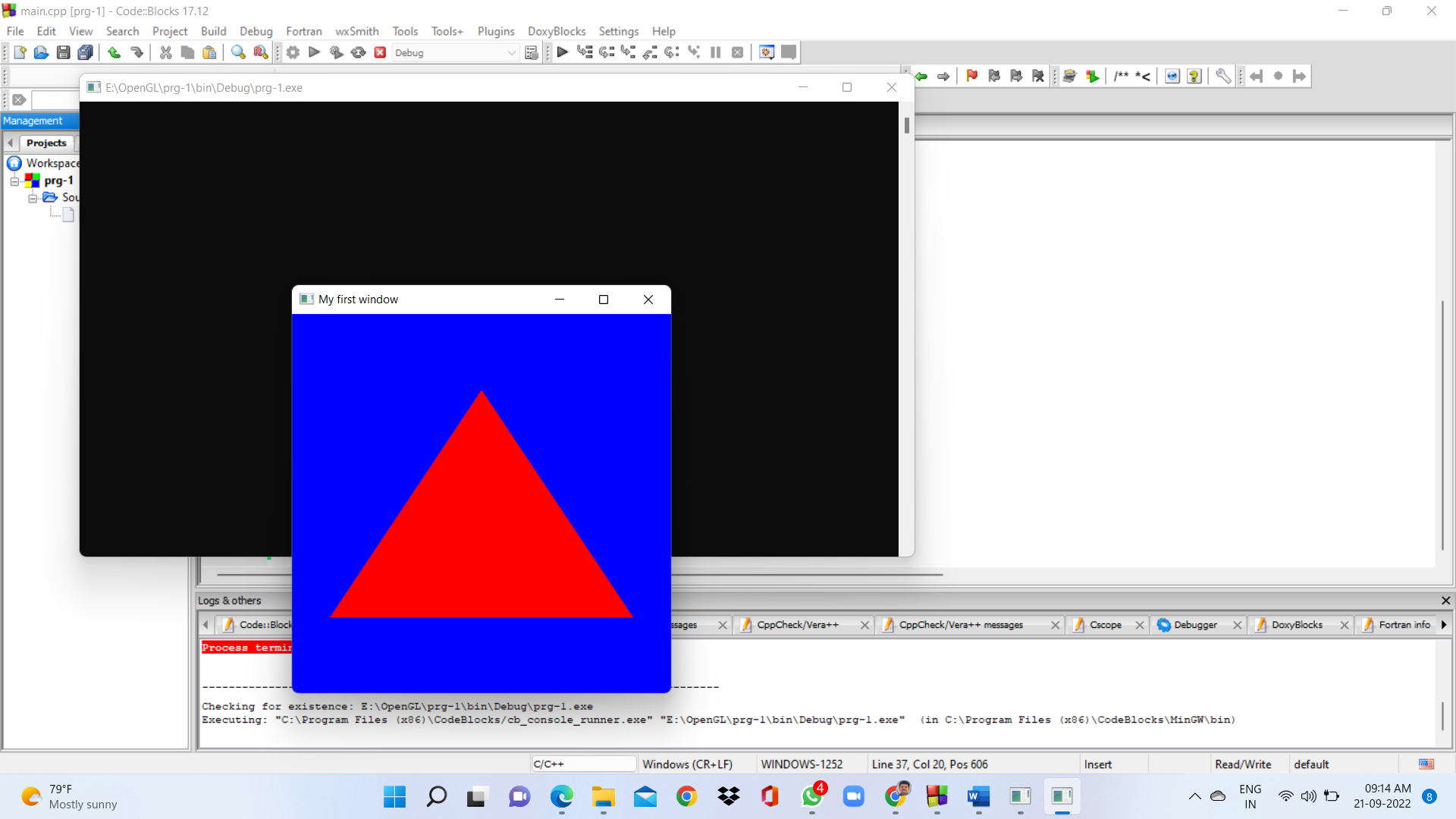
MyInit();

glutDisplayFunc(draw);

glutMainLoop();

return 0;

}



9. Program to draw a line with DDA algorithm.

#include<GL/glut.h>

#include<stdlib.h>

#include<stdio.h>

float x1,x2,y1,y2;

void display(void)

{

float dy,dx,step,x,y,k,Xin,Yin;

dx=x2-x1;

dy=y2-y1;

if(abs(dx)> abs(dy))

{

step = abs(dx);

}

else

step = abs(dy);

Xin = dx/step;

Yin = dy/step;

x= x1;

y=y1;

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

for (k=1 ;k<=step;k++)

{

x= x + Xin;

y= y + Yin;

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

glFlush();

}

void init(void)

{

glClearColor(0.7,0.7,0.7,0.7);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(-100,100,-100,100);

}

int main(int argc, char\*\* argv) {

printf("Enter the value of x1 : ");

scanf("%f",&x1);

printf("Enter the value of y1 : ");

scanf("%f",&y1);

printf("Enter the value of x2 : ");

scanf("%f",&x2);

printf("Enter the value of y2 : ");

scanf("%f",&y2);

glutInit(&argc, argv);

glutInitDisplayMode (GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize (500, 500);

glutInitWindowPosition (100,100);

glutCreateWindow ("DDA Line Algo");

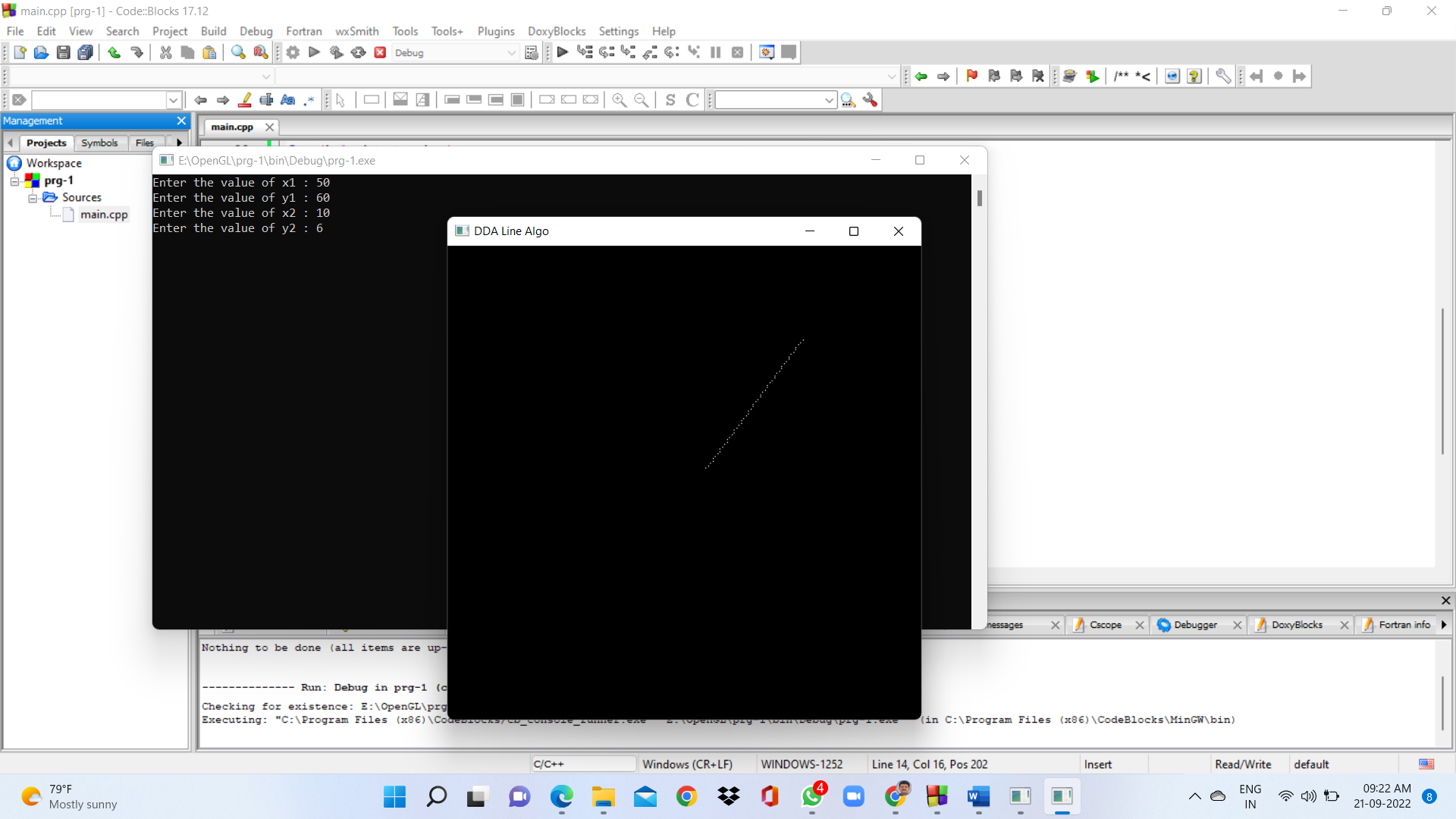
init();

glutDisplayFunc(display);

glutMainLoop();

return 0;

}



10. Program to draw a polygon and transform from its original shape to another using OpenGL.

#include "gl/glut.h"

#include <gl/gl.h>

void Display(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glLoadIdentity();

gluLookAt(0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

glColor3f(0.0, 1.0, 0.0);

glBegin(GL\_POLYGON);

glVertex3f( 0.0, 0.0, 0.0); // V0 ( 0, 0, 0)

glVertex3f( 1.0f, 0.0, 0.0); // V1 ( 1, 0, 0)

glVertex3f( 1.0f, 1.0f, 0.0); // V2 ( 1, 1, 0)

glVertex3f( 0.5f, 1.5f, 0.0); // V3 (0.5, 1.5, 0)

glVertex3f( 0.0, 1.0f, 0.0); // V4 ( 0, 1, 0)

glEnd();

glPushMatrix();

glTranslatef(1.5, 2.0, 0.0);

glRotatef(90.0, 0.0, 0.0, 1.0);

glScalef(0.5, 0.5, 0.5);

glBegin(GL\_POLYGON);

glVertex3f( 0.0, 0.0, 0.0); // V0 ( 0, 0, 0)

glVertex3f( 1.0f, 0.0, 0.0); // V1 ( 1, 0, 0)

glVertex3f( 1.0f, 1.0f, 0.0); // V2 ( 1, 1, 0)

glVertex3f( 0.5f, 1.5f, 0.0); // V3 (0.5, 1.5, 0)

glVertex3f( 0.0, 1.0f, 0.0); // V4 ( 0, 1, 0)

glEnd();

glPopMatrix();

glFlush();

glutSwapBuffers();

}

void Init(void)

{

glClearColor(0.0, 0.0, 0.0, 0.0);

}

void Resize(int width, int height)

{

glViewport(0, 0, width, height);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(60.0, width/height, 0.1, 1000.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

int main(int argc, char \*\*argv)

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowSize(400, 400);

glutInitWindowPosition(200, 200);

glutCreateWindow("Polygon in OpenGL");

Init();

glutDisplayFunc(Display);

glutReshapeFunc(Resize);

glutMainLoop();

return 0;

}

